

FFORMS: Feature-based forecast model selection

Abstract

Features of time series are useful in identifying suitable models for forecasting. We present a general framework, labelled FFORMS (Feature-based FOREcast Model Selection), which selects forecast models based on features calculated from the time series. The FFORMS framework builds a mapping that relates the features of a time series to the “best” forecast model using a random forest. The framework is evaluated using time series from the M1 and M3 competitions and is shown to yield accurate forecasts comparable to several benchmarks and other commonly used automated approaches of time series forecasting. Furthermore, we explore what is happening under the hood of the FFORMS framework. This is accomplished using model-agnostic machine learning interpretability approaches. The analysis provides a valuable insight into how different features and their interactions affect the choice of forecast model.

Keywords: Algorithm selection problem, Time series, Random forest, Machine learning interpretability

1 Introduction

Forecasting is a key activity for any business to operate efficiently. The rapid advances in computing technologies have enabled businesses to keep track of large numbers of time series. Hence, it is becoming increasingly common to have to regularly forecast many millions of time series. For example, large scale businesses may be interested in forecasting sales, cost, and demand for their thousands of products across various locations, warehouses, etc. Technology companies such as Google collect many millions of daily time series such as web-click logs, web search counts, queries, revenues, number of users for different services, etc. Such large collections of time series require fast automated procedures generating accurate forecasts. The scale of these tasks has raised some computational challenges we seek to address by proposing a new fast algorithm for model selection and time series forecasting.

Two alternative strategies for generating such a large number of forecasts are: (1) to either use a single forecasting method across all the time series; or (2) to select an appropriate forecasting

method for each time series individually. It is highly unlikely that a single method will consistently outperform judiciously chosen competitor methods across all time series. We therefore reject the former strategy and focus on an approach for selecting an individual forecasting method for each time series.

Selecting the most appropriate model for forecasting a given time series can be challenging. Two of the most commonly used automated algorithms are the exponential smoothing (ets) algorithm of Hyndman et al. (2002) and the ARIMA (auto.arima) algorithm of Hyndman & Khandakar (2008). Both algorithms are implemented in the forecast package in R (R Core Team 2018; Hyndman et al. 2018). In this paradigm, a class of models is selected in advance, and many models within that class are estimated for each time series. The model with the smallest AICc value is chosen and used for forecasting. This approach relies on the expert judgement of the forecaster in first selecting the most appropriate class of models to use, as it is not usually possible to compare AICc values between model classes due to differences in the way the likelihood is computed, and the way initial conditions are handled.

An alternative approach, which avoids selecting a class of models *a priori*, is to use a simple “hold-out” test set; but then there is often insufficient data to draw a reliable conclusion. To overcome this drawback, time series cross-validation can be used (Hyndman & Athanasopoulos 2018); then models from many different classes may be applied, and the model with the lowest cross-validated MSE selected. However, this increases the computation time involved considerably (at least to order n^2 where n is the number of series to be forecast).

Clearly, there is a need for a fast and scalable algorithm to automate the process of selecting models with the aim of forecasting. There have been several recent studies on the use of meta-learning approaches to automate forecast model selection based on features computed from the time series (Shah 1997; Prudêncio & Ludermir 2004; Lemke & Gabrys 2010; Kück, Crone & Freitag 2016). A meta-learning approach provides systematic guidance on model selection based on knowledge acquired from historical data sets. The key idea is that forecast model selection is posed as a supervised learning task. Each time series in the metadata set is represented as a vector of features and labelled according to the best forecast model (for example, the model with the lowest mean absolute scaled error (MASE) over a test set). Then a meta-learner is trained to identify a suitable forecast model (usually a machine learning algorithm is used). In the era of big data, such an automated model selection process is necessary because the cost of invoking all possible forecast models is prohibitive. However, the existing literature suffers from limited answers to questions such as: How are features related to the property being

modelled? How do features interact with each other to identify a suitable forecast model? Which features contribute most to the classification process? Addressing such questions can enhance the understanding of the relations between features and model selection outcomes. To the best of our knowledge, a very limited effort has been made to understand how the meta-learners are making their decisions and what is really happening inside these complex model structures. Providing transparency will result, building trust in the prediction results of the meta-learner.

We propose a general meta-learning framework using features of the time series to select the class of models, or even the specific model, to be used for forecasting. We refer to this general framework as FFORMS (Feature-based **FOR**ecast-**M**odel **S**election). The forecast model selection process is carried out using a classification algorithm — we use the time series features as inputs, and the “best” forecasting model as the output. A random forest is used to model the relationship between features and best performing forecast model. The classifier is built using a large historical collection of time series, in advance of the forecasting task at hand. Hence, this is an “offline” procedure. The “online” process of generating forecasts only involves calculating the features of a time series and using the pre-trained classifier to identify the best forecasting model. Hence, generating forecasts only involves the estimation of a single forecasting model, with no need to estimate large numbers of models within a class, or to carry out a computationally-intensive cross-validation procedure. To fill the gaps in the literature, this paper makes a first step towards providing a comprehensive analysis of the relationship between time series features and forecast model selection using machine learning interpretability techniques.

In this article, we make the following contributions:

1. We explore the use of random forest algorithm to model the relationship between time series features and the ‘best’ forecast model. The motivation behind the use of random forest classifiers are: i) it can model complex interactions between features; ii) the modelling framework captures linear and non-linear effects of features through the averaging of large number of decision trees; iii) its robustness against over-fitting the training data; iv) it is easy to handle the problem of imbalanced classes; v) it is a fast approach compared to boosting algorithms and vi) it is fairly easy and straightforward to implement with available software. The contribution of this paper differs from previously published work related to meta-learning (Prudêncio & Ludermir 2004; Lemke & Gabrys 2010; Kück, Crone & Freitag 2016) in three ways: i) a more extensive collection of features is used (features to handle multiple seasonality in hourly and daily series), ii) the diversity of forecast models

considered class labels, and iii) the capability of handling high frequency data.

2. The second contribution of the paper is the exploration of what is happening under the hood of the FFORMS framework, leading to an understanding of the relationship between features of time series and the choice of forecast model selection using the FFORMS framework. We call it 'Peeking inside FFORMS'. We try to answer the following questions:
 - i) **Which** features are the most important?
 - ii) **Where** (overall classification or only within specific classes) are they important?
 - iii) **How** are they important?
 - iv) **When** and **how** are features linked to the prediction outcome?
 - v) **When** and **how strongly** do features interact with other features?

2 Literature review

2.1 Time series features

Rather than work with the time series directly at the level of individual observations, we propose analysing time series via an associated "feature space". A time series feature is any measurable characteristic of a time series. For example, [Figure 1](#) shows the time-domain representation of six time series taken from the M3 competition (Makridakis & Hibon 2000) while [Figure 2](#) shows a feature-based representation of the same time series. Here only two features are considered: the strength of seasonality and the strength of trend, calculated based on the measures introduced by Wang, Smith-Miles & Hyndman (2009). Time series in the lower right quadrant of [Figure 2](#) are non-seasonal but trended, while there is only one series with both high trend and high seasonality. We also see how the degree of seasonality and trend varies between series. Other examples of time series features include autocorrelation, spectral entropy and measures of self-similarity and nonlinearity. Fulcher & Jones (2014) identified 9000 operations to extract features from time series.

The choice of the most appropriate set of features depends on both the nature of the time series being analysed, and the purpose of the analysis. In ??, we study the time series from the M1 and M3 competitions (Makridakis et al. 1982; Makridakis & Hibon 2000), and we select features for the purpose of forecast-model selection. The M1 and M3 competitions involve time series of differing length, scale and other properties. We include length as one of our features, but the remaining features are independent of scale and asymptotically independent of the length of the time series (i.e., they are ergodic). As our main focus is forecasting, we select features which

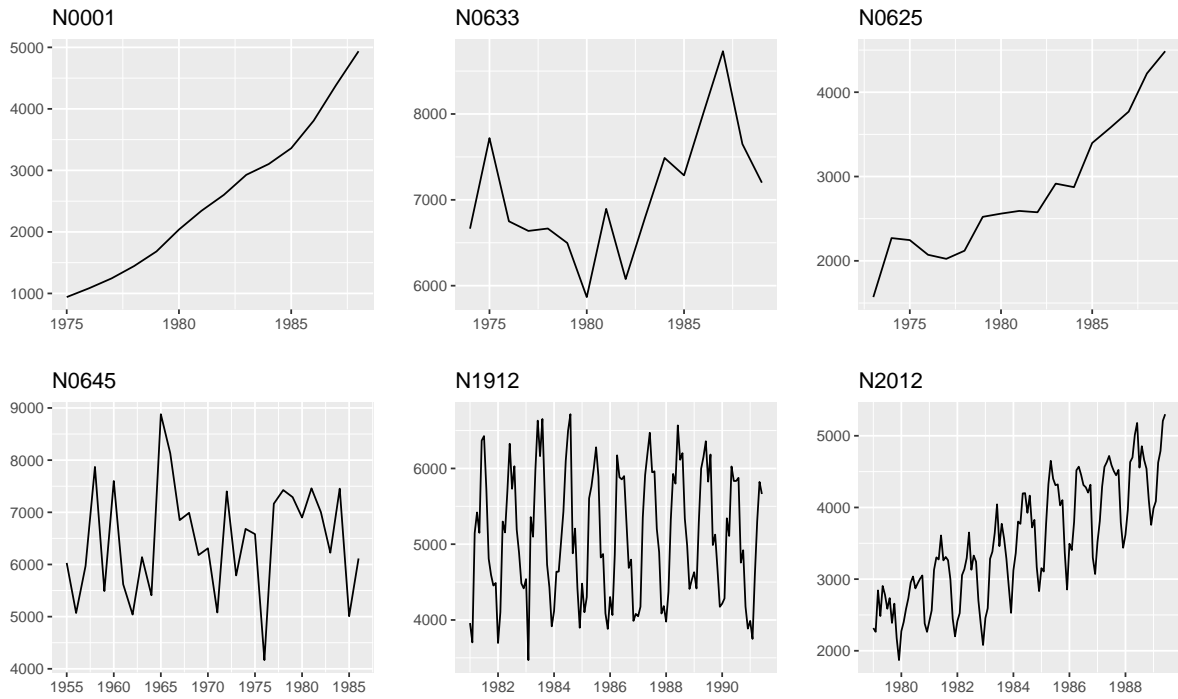


Figure 1: Time-domain representation of time series

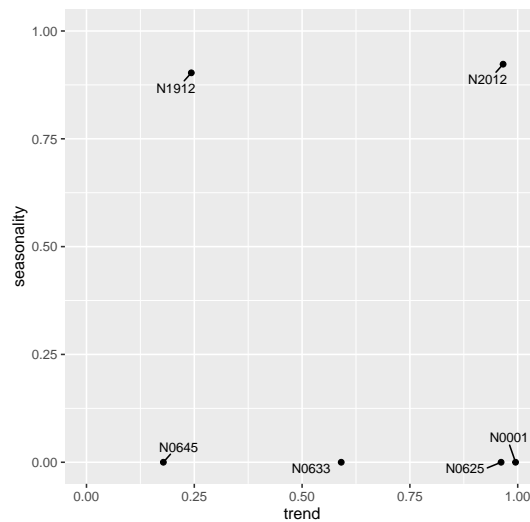


Figure 2: Feature-based representation of time series

have discriminatory power in selecting a good model for forecasting.

2.2 Meta-learning for algorithm selection

John Rice was an early and strong proponent of the idea of meta-learning, which he called the algorithm selection problem (ASP) (Rice 1976). The term *meta-learning* started to appear with the emergence of the machine-learning literature. Rice's framework for algorithm selection is shown in Figure 3 and comprises four main components. The problem space, P , represents the data sets used in the study. The feature space, F , is the range of measures that characterize

the problem space P . The algorithm space, A , is a list of suitable candidate algorithms which can be used to find solutions to the problems in P . The performance metric, Y , is a measure of algorithm performance such as accuracy, speed, etc. A formal definition of the algorithm selection problem is given by Smith-Miles (2009), and repeated below.

Algorithm selection problem. For a given problem instance $x \in P$, with features $f(x) \in F$, find the selection mapping $S(f(x))$ into algorithm space A , such that the selected algorithm $\alpha \in A$ maximizes the performance mapping $y(\alpha(x)) \in Y$.

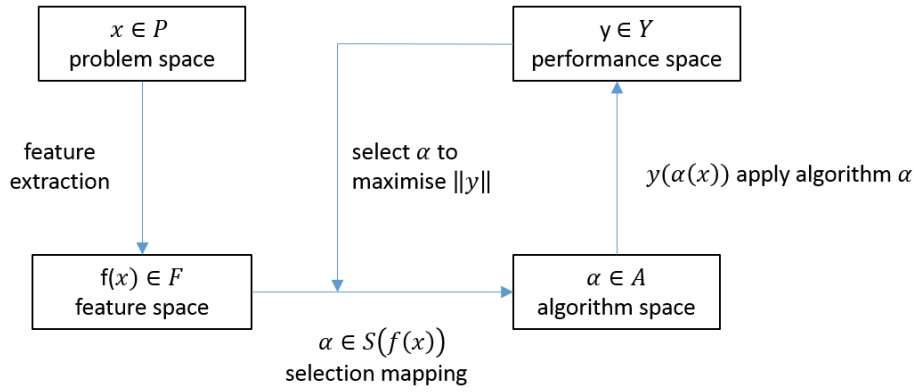


Figure 3: Rice's framework for the Algorithm Selection Problem.

The main challenge in ASP is to identify the selection mapping S from the feature space to the algorithm space. Even though Rice's framework articulates a conceptually rich framework, it does not specify how to obtain S . This gives rise to the meta-learning approach.

2.3 Forecast-model selection using meta-learning

Selecting models for forecasting can be framed according to Rice's ASP framework.

Forecast-model selection problem. For a given time series $x \in P$, with features $f(x) \in F$, find the selection mapping $S(f(x))$ into the algorithm space A , such that the selected algorithm $\alpha \in A$ minimizes forecast accuracy error metric $y(\alpha(x)) \in Y$ on the test set of the time series.

Existing methods differ with respect to the way they define the problem space (A), the features (F), the forecasting accuracy measure (Y) and the selection mapping (S).

Collopy & Armstrong (1992) introduced 99 rules based on 18 features of time series, in order to make forecasts for economic and demographic time series. This work was extended by Armstrong (2001) to reduce human intervention.

Shah (1997) used the following features to classify time series: the number of observations, the ratio of the number of turning points to the length of the series, the ratio of number of step changes, skewness, kurtosis, the coefficient of variation, autocorrelations at lags 1–4, and partial autocorrelations at lag 2–4. Casting Shah’s work in Rice’s framework, we can specify: $P = 203$ quarterly series from the M1 competition (Makridakis et al. 1982); $A = 3$ forecasting methods, namely simple exponential smoothing, Holt-Winters exponential smoothing with multiplicative seasonality, and a basic structural time series model; $Y =$ mean squared error for a hold-out sample. Shah (1997) learned the mapping S using discriminant analysis.

Prudêncio & Ludermir (2004) was the first paper to use the term “meta-learning” in the context of time series model selection. They studied the applicability of meta-learning approaches for forecast-model selection based on two case studies. Again using the notation above, we can describe their first case study with: A contained only two forecasting methods, simple exponential smoothing and a time-delay neural network; $Y =$ mean absolute error; F consisted of 14 features, namely length, autocorrelation coefficients, coefficient of variation, skewness, kurtosis, and a test of turning points to measure the randomness of the time series; S was learned using the C4.5 decision tree algorithm. For their second study, the algorithm space included a random walk, Holt’s linear exponential smoothing and AR models; the problem space P contained the yearly series from the M3 competition (Makridakis & Hibon 2000); F included a subset of features from the first study; and Y was a ranking based on error. Beyond the task of forecast-model selection, they used the NOEMON approach to rank the algorithms (Kalousis & Theoharis 1999).

Lemke & Gabrys (2010) studied the applicability of different meta-learning approaches for time series forecasting. Their algorithm space A contained ARIMA models, exponential smoothing models and a neural network model. In addition to statistical measures such as the standard deviation of the de-trended series, skewness, kurtosis, length, strength of trend, Durbin-Watson statistics of regression residuals, the number of turning points, step changes, a predictability measure, nonlinearity, the largest Lyapunov exponent, and auto-correlation and partial-autocorrelation coefficients, they also used frequency domain based features. The feed forward neural network, decision tree and support vector machine approaches were considered to learn the mapping S .

Wang, Smith-Miles & Hyndman (2009) used a meta-learning framework to provide recommendations as to which forecast method to use to generate forecasts. In order to evaluate forecast accuracy, they introduced a new measure $Y = \text{simple percentage better (SPB)}$, which calculates the

forecasting accuracy of a method against the forecasting accuracy error of random walk model. They used a feature set F comprising nine features: strength of trend, strength of seasonality, serial correlation, nonlinearity, skewness, kurtosis, self-similarity, chaos and periodicity. The algorithm space A included eight forecast-models, namely, exponential smoothing, ARIMA, neural networks and random walk model; while the mapping S was learned using the C4.5 algorithm for building decision trees. In addition, they used SOM clustering on the features of the time series in order to understand the nature of time series in a two-dimensional setting.

The set of features introduced by Wang, Smith-Miles & Hyndman (2009) was later used by Widodo & Budi (2013) to develop a meta-learning framework for forecast-model selection. The authors further reduced the dimensionality of time series by performing principal component analysis on the features.

More recently, Kück, Crone & Freitag (2016) proposed a meta-learning framework based on neural networks for forecast-model selection. Here, $P = 78$ time series from the NN3 competition were used to build the meta-learner. They introduced a new set of features based on forecasting errors. The average symmetric mean absolute percentage error was used to identify the best forecast-models for each series. They classify their forecast-models in the algorithm space A , comprising single, seasonal, seasonal-trend and trend exponential smoothing. The mapping S was learned using a feed-forward neural network. Further, they evaluated the performance of different sets of features for forecast-model selection.

Appendix A: Definition of features

Length of time series

The appropriate forecast method for a time series depends on how many observations are available. For example, shorter series tend to need simple models such as a random walk. On the other hand, for longer time series, we have enough information to be able to estimate a number of parameters. For even longer series (over 200 observations), models with time-varying parameters give good forecasts as they help to capture the changes of the model over time.

Features based on an STL-decomposition

The strength of trend, strength of seasonality, linearity, curvature, spikiness and first autocorrelation coefficient of the remainder series, are calculated based on a decomposition of the time series. Suppose we denote our time series as y_1, y_2, \dots, y_T . First, an automated Box-Cox transformation (Guerrero 1993) is applied to the time series in order to stabilize the variance and

to make the seasonal effect additive. The transformed series is denoted by y_t^* . For quarterly and monthly data, this is decomposed using STL (Cleveland, Cleveland & Terpenning 1990) to give $y_t^* = T_t + S_t + R_t$, where T_t denotes the trend, S_t denotes the seasonal component, and R_t is the remainder component. For non-seasonal data, Friedman's super smoother (Friedman 1984) is used to decompose $y_t^* = T_t + R_t$, and $S_t = 0$ for all t . The de-trended series is $y_t^* - T_t = S_t + R_t$, the deseasonalized series is $y_t^* - S_t = T_t + R_t$.

The strength of trend is measured by comparing the variance of the deseasonalized series and the remainder series (Wang, Smith-Miles & Hyndman 2009):

$$\text{Trend} = \max [0, 1 - \text{Var}(R_t) / \text{Var}(T_t + R_t)] .$$

Similarly, the strength of seasonality is computed as

$$\text{Seasonality} = \max [0, 1 - \text{Var}(R_t) / \text{Var}(S_t + R_t)] .$$

The linearity and curvature features are based on the coefficients of an orthogonal quadratic regression

$$T_t = \beta_0 + \beta_1 \phi_1(t) + \beta_2 \phi_2(t) + \varepsilon_t,$$

where $t = 1, 2, \dots, T$, and ϕ_1 and ϕ_2 are orthogonal polynomials of orders 1 and 2. The estimated value of β_1 is used as a measure of linearity while the estimated value of β_2 is considered as a measure of curvature. These features were used by Hyndman, Wang & Laptev (2015). The linearity and curvature depend on the the scale of the time series. Therefore, the time series are scaled to mean zero and variance one before these two features are computed.

The spikiness feature is useful when a time series is affected by occasional outliers. Hyndman, Wang & Laptev (2015) introduced an index to measure spikiness, computed as the variance of the leave-one-out variances of r_t .

We compute the first autocorrelation coefficient of the remainder series, r_t .

Stability and lumpiness

The features "stability" and "lumpiness" are calculated based on tiled windows (i.e., they do not overlap). For each window, the sample mean and variance are calculated. The stability feature is calculated as the variance of the means, while lumpiness is the variance of the variances. These were first used by Hyndman, Wang & Laptev (2015).

Spectral entropy of a time series

Spectral entropy is based on information theory, and can be used as a measure of the forecastability of a time series. Series that are easy to forecast should have a small spectral entropy value, while very noisy series will have a large spectral entropy. We use the measure introduced by Goerg (2013) to estimate the spectral entropy. It estimates the Shannon entropy of the spectral density of the normalized spectral density, given by

$$H_s(y_t) := - \int_{-\pi}^{\pi} \hat{f}_y(\lambda) \log \hat{f}_y(\lambda) d\lambda,$$

where \hat{f}_y denotes the estimate of the spectral density introduced by Nuttall & Carter (1982). The R package ForeCA (Goerg 2016) was used to compute this measure.

Hurst exponent

The Hurst exponent measures the long-term memory of a time series. The Hurst exponent is given by $H = d + 0.5$, where d is the fractal dimension obtained by estimating a ARFIMA(0, d , 0) model. We compute this using the maximum likelihood method (Haslett & Raftery 1989) as implemented in the fracdiff package (Fraley 2012). This measure was also used in Wang, Smith-Miles & Hyndman (2009).

Nonlinearity

To measure the degree of nonlinearity of the time series, we use statistic computed in Terasvirta's neural network test for nonlinearity (Teräsvirta, Lin & Granger 1993), also used in Wang, Smith-Miles & Hyndman (2009). This takes large values when the series is nonlinear, and values around 0 when the series is linear.

Parameter estimates of an ETS model

The ETS(A,A,N) model (Hyndman et al. 2008) produces equivalent forecasts to Holt's linear trend method, and can be expressed as follows:

$$\begin{aligned} y_t &= \ell_{t-1} + b_{t-1} + \varepsilon_t \\ \ell_t &= \ell_{t-1} + b_{t-1} + \alpha \varepsilon_t \\ b_t &= b_{t-1} + \beta \varepsilon_t, \end{aligned}$$

where α is the smoothing parameter for the level, and β is the smoothing parameter for the trend. We include the parameter estimates of both α and β in our feature set for yearly time series. These indicate the variability in the level and slope of the time series.

The ETS(A,A,A) model (Hyndman et al. 2008) produces equivalent forecasts to Holt-Winters' additive method, and can be expressed as follows:

$$\begin{aligned}y_t &= \ell_{t-1} + b_{t-1} + s_{t-m} + \varepsilon_t \\ \ell_t &= \ell_{t-1} + b_{t-1} + s_{t-m} + \alpha \varepsilon_t \\ b_t &= b_{t-1} + \beta \varepsilon_t, \\ s_t &= s_{t-m} + \gamma \varepsilon_t,\end{aligned}$$

where γ is the smoothing parameter for the seasonal component, and the other parameters are as above. We include the parameter estimates of α , β and γ in our feature set for monthly and quarterly time series. The value of γ provides a measure for the variability of the seasonality of a time series.

Unit root test statistics

The Phillips-Perron test is based on the regression $y_t = c + \alpha y_{t-1} + \varepsilon_t$. The test statistic we use as a feature is the usual “Z-alpha” statistic with the Bartlett window parameter set to the integer value of $4(T/100)^{0.25}$ (Pfaff 2008). This is the default value returned from the `ur.pp()` function in the `urca` package (Pfaff, Zivot & Stigler 2016).

The KPSS test is based on the regression $y_t = c + \delta t + \alpha y_{t-1} + \varepsilon_t$. The test statistic we use as a feature is the usual KPSS statistic with the Bartlett window parameter set to the integer value of $4(T/100)^{0.25}$ (Pfaff 2008). This is the default value returned from the `ur.kpss()` function in the `urca` package (Pfaff, Zivot & Stigler 2016).

Autocorrelation coefficient based features

We calculate the first-order autocorrelation coefficient and the sum of squares of the first five autocorrelation coefficients of the original series, the first-differenced series, the second-differenced series, and the seasonally differenced series (for seasonal data).

A linear trend model is fitted to the time series, and the first-order autocorrelation coefficient of the residual series is calculated.

We calculate the sum of squares of the first five partial autocorrelation coefficients of the original series, the first-differenced series and the second-differenced series.

References

- Armstrong, JS (2001). Should we redesign forecasting competitions? *International Journal of Forecasting* **17**(1), 542–543.
- Cleveland, RB, WS Cleveland & I Terpenning (1990). STL: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics* **6**(1), 3.
- Collopy, F & JS Armstrong (1992). Rule-based forecasting: development and validation of an expert systems approach to combining time series extrapolations. *Management Science* **38**(10), 1394–1414.
- Fraley, C (2012). *fracdiff: Fractionally differenced ARIMA aka ARFIMA(p,d,q) models*. R package version 1.4-2. <https://CRAN.R-project.org/package=fracdiff>.
- Friedman, JH (1984). *A variable span scatterplot smoother*. Technical Report 5. Laboratory for Computational Statistics, Stanford University.
- Fulcher, BD & NS Jones (2014). Highly comparative feature-based time-series classification. *IEEE Transactions on Knowledge and Data Engineering* **26**(12), 3026–3037.
- Goerg, GM (2016). *ForeCA: An R package for forecastable component analysis*. R package version 0.2.4. <https://CRAN.R-project.org/package=ForeCA>.
- Goerg, G (2013). Forecastable component analysis. In: *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pp.64–72.
- Guerrero, VM (1993). Time-series analysis supported by power transformations. *Journal of Forecasting* **12**, 37–48.
- Haslett, J & AE Raftery (1989). Space-time modelling with long-memory dependence: Assessing Ireland’s wind power resource. *Applied Statistics* **38**(1), 1–50.
- Hyndman, RJ & G Athanasopoulos (2018). *Forecasting: principles and practice*. 2nd ed. Melbourne, Australia: OTexts. <https://OTexts.org/fpp2/>.
- Hyndman, RJ & Y Khandakar (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software* **26**(3), 1–22. <http://www.jstatsoft.org/article/view/v027i03>.
- Hyndman, RJ, AB Koehler, JK Ord & RD Snyder (2008). *Forecasting with exponential smoothing: the state space approach*. Berlin: Springer.
- Hyndman, RJ, AB Koehler, RD Snyder & S Grose (2002). A state space framework for automatic forecasting using exponential smoothing methods. *International Journal of Forecasting* **18**(3), 439–454.
- Hyndman, RJ, E Wang & N Laptev (2015). Large-scale unusual time series detection. In: *Data Mining Workshop (ICDMW), 2015 IEEE International Conference on*. IEEE, pp.1616–1619.

- Hyndman, R, G Athanasopoulos, C Bergmeir, G Caceres, L Chhay, M O'Hara-Wild, F Petropoulos, S Razbash, E Wang & F Yasmeen (2018). *forecast: Forecasting functions for time series and linear models*. R package version 8.3. <http://pkg.robjhyndman.com/forecast>.
- Kalousis, A & T Theoharis (1999). NOEMON: Design, implementation and performance results of an intelligent assistant for classifier selection. *Intelligent Data Analysis* **3**(5), 319–337.
- Kück, M, SF Crone & M Freitag (2016). Meta-learning with neural networks and landmarking for forecasting model selection an empirical evaluation of different feature sets applied to industry data. In: *Neural Networks (IJCNN), 2016 International Joint Conference on*. IEEE, pp.1499–1506.
- Lemke, C & B Gabrys (2010). Meta-learning for time series forecasting and forecast combination. *Neurocomputing* **73**(10), 2006–2016.
- Makridakis, S, A Andersen, R Carbone, R Fildes, M Hibon, R Lewandowski, J Newton, E Parzen & R Winkler (1982). The accuracy of extrapolation (time series) methods: results of a forecasting competition. *Journal of forecasting* **1**(2), 111–153.
- Makridakis, S & M Hibon (2000). The M3-Competition: results, conclusions and implications. *International Journal of Forecasting* **16**(4), 451–476.
- Nuttall, AH & GC Carter (1982). Spectral estimation using combined time and lag weighting. *Proceedings of the IEEE* **70**(9), 1115–1125.
- Pfaff, B (2008). *Analysis of integrated and cointegrated time series with R*. Springer.
- Pfaff, B, E Zivot & M Stigler (2016). *urca: Unit root and cointegration tests for time series data*. R package version 1.3-0. <https://CRAN.R-project.org/package=urca>.
- Prudêncio, RB & TB Ludermir (2004). Meta-learning approaches to selecting time series models. *Neurocomputing* **61**, 121–137.
- R Core Team (2018). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. <https://www.R-project.org/>.
- Rice, JR (1976). The algorithm selection problem. *Advances in Computers* **15**, 65–118.
- Shah, C (1997). Model selection in univariate time series forecasting using discriminant analysis. *International Journal of Forecasting* **13**(4), 489–500.
- Smith-Miles, K (2009). Cross-disciplinary perspectives on meta-learning for algorithm selection. *ACM Computing Surveys (CSUR)* **41**(1), 6.
- Teräsvirta, T, CF Lin & CWJ Granger (1993). Power of the neural network linearity test. *Journal of Time Series Analysis* **14** (2), 209–220.
- Wang, X, K Smith-Miles & RJ Hyndman (2009). Rule induction for forecasting method selection: meta-learning the characteristics of univariate time series. *Neurocomputing* **72**(10), 2581–2594.

Widodo, A & I Budi (2013). "Model selection using dimensionality reduction of time series characteristics". Paper presented at the International Symposium on Forecasting, Seoul, South Korea. June 2013. <https://goo.gl/ig2J57>.